First Hit

☐ Generate Collection Print

L3: Entry 8 of 174 File: PGPB Feb 7, 2002

DOCUMENT-IDENTIFIER: US 20020015039 A1
TITLE: Rendering graphic object based images

Abstract Paragraph:
Disclosed are methods, apparatus (1) and computer readable medium for generating instructions for a directed adjacency graph, such as an expression tree, into a raster pixel image having a plurality of scan lines and a plurality of pixel locations on each scan line. The expression tree comprises one or more parent nodes and one or more leaf nodes. The parent nodes each representing an operator and each having branches to respective descendent nodes. The leaf nodes each representing a graphic object. The apparatus comprises means for scanning a plurality of pixel locations (300). The apparatus further comprising a module (504) for determining, for each of the scanned pixel locations, a portion of the expression tree in accordance with the activity of the operators, where the portion of the expression tree is that portion which passes data up the tree. The apparatus also comprises a Module (506) for generating instructions for the determined portion of the expression tree, wherein operator instructions are generated for those operators of the determined portion of the expression tree having active branches and wherein leaf instructions are generated for those graphic objects which are active at the scanned pixel location.

Application Filing Date:
20010418

Summary of Invention Paragraph:
[0009] However, these priority encoders lack sufficient flexibility to cope with a tree based graphical expression. In particular, those expressions which include OUT and ATOP Porter and Duff compositing operators as described in "Compositing Digital Images", Porter, T; Duff, T; Computer Graphics, Vol. 18 No. 3 (1984) pp. 253-259. The problems arise because the compositing operation depends on which of the graphic objects composited by the operation are active at a given pixel location. One solution to this problem is to use a complex arrangement of clipping objects, which requires a lot of extra edge processing and requires a large number of levels for the clipping objects.

Summary of Invention Paragraph:
[0011] According to a first aspect of the invention, there is provided a method of generating instructions for a directed adjacency graph, said directed adjacency graph comprising one or more parent nodes and one or more leaf nodes, each of which said parent node representing an operator and having branches to respective descendent nodes, and each of which said leaf node representing a graphic object, said method comprising the steps of: determining groups of one or more pixel locations; determining, for each said group, a portion of said directed adjacency graph in accordance with activities of the operators, wherein the said portion of the directed adjacency graph is that portion which passes data up the directed adjacency graph; and generating, for each said group, instructions for the determined portion of the directed adjacency graph, wherein operator instructions are generated for those operators of the determined portion of the directed adjacency graph having active branches and wherein leaf instructions are generated for those graphic objects which are active at said group of one or more pixel

locations.

Summary of Invention Paragraph:
[0012] According to a second aspect of the invention, there is provided a method of generating instructions for an expression tree, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said method comprising the steps of: determining groups of one or more pixel locations; determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; traversing, for each said group, said expression tree, wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; and generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node.

Summary of Invention Paragraph:
[0013] According to a third aspect of the invention, there is provided a method of rendering an expression tree into a raster pixel image having a plurality of scanlines and a plurality of pixel locations on each said scanline, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said method comprising the steps of: generating a table representative of said expression tree, wherein said table comprises a plurality of records corresponding to associated said binary nodes and leaf nodes, and each said record of a said associated binary node comprises a first field indicating whether a said left region is required for operation of the operator of said associated binary node, a second field indicating whether a right region is required for operation of the operator of said associated binary node, a third field capable of indicating whether a said left branch of said associated binary node is active, and a fourth field capable of indicating whether a said right branch of said associated binary node is active; determining groups of one or more pixel locations; determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; updating, for each said group, said third and fourth fields of said table for said determined active and inactive branches; traversing, for each said group, said expression tree in accordance with said updated table wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said

right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node; and executing, for each said group, corresponding said generated instructions so as to render said image.

Summary of Invention Paragraph:
[0014] According to a fourth aspect of the invention, there is provided apparatus for generating instructions for a directed adjacency graph, said directed adjacency graph comprising one or more parent nodes and one or more leaf nodes, each of which said parent node representing an operator and having branches to respective descendent nodes, and each of which said leaf node representing a graphic object, said apparatus comprising: means for determining groups of one or more pixel locations; means for determining, for each said group, a portion of said directed adjacency graph in accordance with activities of the operators, wherein the said portion of the directed adjacency graph is that portion which passes data up the directed adjacency graph; and means for generating, for each said group, instructions for the determined portion of the directed adjacency graph, wherein operator instructions are generated for those operators of the determined portion of the directed adjacency graph having active branches and wherein leaf instructions are generated for those graphic objects which are active at said group of one or more pixel locations.

Summary of Invention Paragraph:
[0015] According to a fifth aspect of the invention, there is provided apparatus for generating instructions for an expression tree, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said apparatus comprising: means for determining groups of one or more pixel locations; means for determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; means for traversing, for each said group, said expression tree, wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; and means for generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node.

Summary of Invention Paragraph:
[0016] According to a sixth aspect of the invention, there is provided apparatus for rendering an expression tree into a raster pixel image having a plurality of scanlines and a plurality of pixel locations on each said scanline, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node

represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said apparatus comprising: means for generating a table representative of said expression tree, wherein said table comprises a plurality of records corresponding to associated said binary nodes and leaf nodes, and each said record of a said associated binary node comprises a first field indicating whether a said left region is required for operation of the operator of said associated binary node, a second field indicating whether a right region is required for operation of the operator of said associated binary node, a third field capable of indicating whether a said left branch of said associated binary node is active, and a fourth field capable of indicating whether a said right branch of said associated binary node is active; means for determining groups of one or more pixel locations; means for determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; means for updating, for each said group, said third and fourth fields of said table for said determined active and inactive branches; means for traversing, for each said group, said expression tree in accordance with said updated table wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; means for generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node; and means for executing, for each said group, corresponding said generated instructions so as to render said image.

Summary of Invention Paragraph:
[0017] According to a seventh aspect of the invention, there is provided a computer readable medium comprising a computer program for generating instructions for a directed adjacency graph, said directed adjacency graph comprising one or more parent nodes and one or more leaf nodes, each of which said parent node representing an operator and having branches to respective descendent nodes, and each of which said leaf node representing a graphic object, said computer program comprising: code for determining groups of one or more pixel locations; code for determining, for each said group, a portion of said directed adjacency graph in accordance with activities of the operators, wherein the said portion of the directed adjacency graph is that portion which passes data up the directed adjacency graph; and code for generating, for each said group, instructions for the determined portion of the directed adjacency graph, wherein operator instructions are generated for those operators of the determined portion of the directed adjacency graph having active branches and wherein leaf instructions are generated for those graphic objects which are active at said group of one or more pixel locations.

Summary of Invention Paragraph:
[0018] According to an eighth aspect of the invention, there is provided a computer readable medium comprising a computer program for generating instructions for an expression tree, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said computer program comprising:

code for determining groups of one or more pixel locations; code for determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; code for traversing, for each said group, said expression tree, wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; and code for generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node.

Summary of Invention Paragraph:
[0019] According to a ninth aspect of the invention, there is provided a computer readable medium comprising a computer program for rendering an expression tree into a raster pixel image having a plurality of scanlines and a plurality of pixel locations on each said scanline, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said computer program comprising: code for generating a table representative of said expression tree, wherein said table comprises a plurality of records corresponding to associated said binary nodes and leaf nodes, and each said record of a said associated binary node comprises a first field indicating whether a said left region is required for operation of the operator of said associated binary node, a second field indicating whether a right region is required for operation of the operator of said associated binary node, a third field capable of indicating whether a said left branch of said associated binary node is active, and a fourth field capable of indicating whether a said right branch of said associated binary node is active; code for determining groups of one or more pixel locations; code for determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; code for updating, for each said group, said third and fourth fields of said table for said determined active and inactive branches; code for traversing, for each said group, said expression tree in accordance with said updated table wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; code for generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node; and code for executing, for each said group, corresponding said generated instructions so as to render said image.

Detail Description Paragraph:
[0062] The operation of the preferred embodiment will be described with reference

to the simple example of rendering an image 78 shown in FIG. 8. The image 78 is seen to include two graphical objects, in particular, a partly transparent blue-coloured triangle 80 rendered on top of and thereby partly obscuring an opaque red coloured rectangle 90. As seen, the rectangle 90 includes side edges 92, 94, 96 and 98 defined between various pixel positions (X) and scan line positions (Y). Because the edges 96 and 98 are formed upon the scan lines (and thus parallel therewith), the actual object description of the rectangle 90 can be based solely upon the side edges 92 and 94, such as seen in FIG. 9A. In this connection, edge 92 commences at pixel location (40,35) and extends in a raster direction down the screen to terminate at pixel position (40,105). Similarly, the edge 94 extends from pixel position (160,35) to position (160,105). The horizontal portions of the rectangular graphic object 90 may be obtained merely by scanning from the edge 92 to the edge 94 in a rasterised fashion.

Detail Description Paragraph:
[0063] The blue triangular object 80 however is defined by three object edges 82, 84 and 86, each seen as vectors that define the vertices of the triangle. Edges 82 and 84 are seen to commence at pixel location (100,20) and extend respectively to pixel locations (170,90) and (30,90). Edge 86 extends between those two pixel locations in a traditional rasterised direction of left to right. In this specific example because the edge 86 is horizontal like the edges 96 and 98 mentioned above, is it not essential that the edge 86 be defined, since the edge 86 is characterised by the related endpoints of the edges 82 and 84. In addition to the starting and ending pixel locations used to describe the edges 82 and 84, each of these edges will have associated therewith the slope value in this case +1 and -1 respectively.

Detail Description Paragraph:
[0148] An example of where the spill list procedure is utilised is seen in FIG. 13A where three arbitrary edges 60, 61 and 63 intersect an arbitrary edge 62 at a relative position between scan lines A and B. Further, the actual displayed pixel locations 64 for each of scan lines A, B, are shown which span pixel locations C to J. In the above described example where the edge pool 412 is size to retain three edge records, it will be apparent that such an arrangement alone will not be sufficient to accommodate three edge intersections occurring between adjacent scan lines as illustrated in FIG. 13A.

Detail Description Paragraph:
[0158] Turning now to FIG. 19, there is shown a typical example of a simple expression tree and a corresponding instruction list. The expression tree comprises five leaf nodes 10, 9, 7, 6 and 0-5 describing the graphic objects A, B, C, D and PAGE respectively. The expression tree also contains nodes 8, 11, 12, and 13 each having two branches and representing the operations "out", "in", "over" and "over" respectively. The instruction list contains a list of instructions for rendering the expression tree. However, problems arise when generating an instruction list for a given pixel location because the compositing operation depends on which of the graphic objects composited by the operation are active at that given pixel location. The preferred Activity Determination and Instruction Generation Module 500 seeks to solve these problems.

Detail Description Paragraph:
[0159] The operation of the Activity Determination and Instruction Generation Module 500 will now be described with reference to FIG. 5. The Instruction Generator 300 during an initialisation phase passes to the Level Activation Table Generator 502 an object graphic description of the image to be rendered. This object graphic description is in the form of an expression tree, such as for example the expression tree shown in FIG. 19. The Level Activation Table Generator 502 then generates a Level Activation Table, which is a linear form of the expression tree and stores it the Level Activation Table Store 34. The generated Level Activation Table contains a plurality of records, one record for each binary

operator node, unary node and leaf node of the expression tree.

Detail Description Paragraph:
[0160] In addition, the Level Activation Table also contains fields for storing data on certain inherent properties of the binary operators, and fields for storing data on the activity of the branches of the binary operator nodes. The data concerning the properties of the binary operators is static, that is dependent on the actual operators used and not on the location of the currently scanned pixel. Consequently, this data can be generated and stored in the level activation table during the initialisation phase. However, the data concerning branch activity is dependent upon the currently scanned pixel location. During the initialisation phase, as there is no currently active pixel location, the branch activity data of the Level Activation Table is initialised as inactive.

Detail Description Paragraph:
[0161] The Pixel Sequential Rendering Apparatus 20 then commences scanning the pixel locations of the image in a raster scan order. The edge processing module 400 passes to the Level Activation Table Update Module 504, the pixel locations of the edges of the current scan line being scanned. The Level Activation Table Update Module 504 accesses the Level Activation Table from the store 34 and updates the Table depending upon which pixel location is currently being scanned. For ease of implementation, the Table is updated only once for those pixel locations between adjacent edges. This is based on the recognition that the instruction list for rendering the expression tree is the same for the group of pixel locations between any two adjacent edges of a scan line. Alternatively, the Table can be updated for each scanned pixel location.

Detail Description Paragraph:
[0162] The Traversal and Instruction Generator Module 506, then generates instructions based on this updated Level Activation Table which instructions are then passed onto the Fill Colour Determination Module 600. This process is continued for each group of pixel locations between adjacent edges until the image is rendered.

Detail Description Paragraph:
[0178] In the example of FIG. 20, the Activity Determination and Instruction Generation Module 500 after the initialisation phase then determines for a particular group of scanned pixel locations, that object C is active and Objects A, B, and D are inactive. The Module 500 determines the activity of the branches of each of the binary nodes from the activity of the objects. Namely, all the branches directly coupling the active leaf node objects (e.g. Object C and PAGE) to the root node (eg. 13) will be active. In this description, an active left branch is designated as L Active=TRUE, an inactive left branch is designated as L Active=FALSE, an active right branch is designated as R Active=TRUE, and an inactive right branch is designated as R Active=FALSE. In the example of FIG. 20, the Module 500 determines that:

Detail Description Paragraph:
[0184] The Activity Determination and Instruction Generation Module 500 then traverses and generates instructions for the current group of scanned pixel locations depending upon of the activity of the binary operators, the activity of the branches of the expression tree, and the activity of the graphic objects (leaf nodes). The Module 500 traverses the expression tree commencing at the root node (eg. node 13) in a top down--left to right manner.

Detail Description Paragraph:
[0189] In the example of FIG. 20, the Activity Determination and Instruction Generation Module 500 begins its traversal at root node 13. As both the left and right branches of the root node 13 are active, the Module 500 generates an "over" operator instruction. For the same reasons, the Module 500 will traverse to binary

node 12 and binary node 0-5. As the Module 500 traverses in a top down left to right manner, it first traverses to binary node 12. At this binary node 12, the Module 500 does not generate an "over" operation, as the left branch of the binary 12 is inactive. For the same reasons, the Module 500 does not traverse the left branch to binary node 11. However, as the right branch of binary node 12 is active and the left branch of binary node 12 is inactive and as ({overscore (L)}.andgate.R)=TRUE for the "over" operator, the Module 500 traverses to binary node 8. At this binary node 8, the Module 500 does not generate an "out" operation, as the right branch of the binary 8 is inactive. For the same reasons, the Module 500 does not traverse the right branch to leaf node 6. However, as the left branch of binary node 8 is active and the right branch of binary node 8 is inactive and as (L.andgate.{overscore (R)})=TRUE for the "out" operator, the Module 500 traverses to leaf node 7, where the Module 500 generates a leaf value instruction C. The Module 500 then returns to root node 13 to traverse to leaf node 0-5, where the Module 500 generates a leaf value instruction PAGE. Comparing FIGS. 19 and 20, it can thus be seen that the Module 500 generates a minimal instruction set corresponding to the expression tree for that current group of scanned pixel locations. The Module 500 will then repeat the operation for the next group of scanned pixel locations between the next adjacent edges.

Detail Description Paragraph:
[0191] As mentioned above, the Level Activation Table Generator 502 generates a Level Activation Table. Turning now to FIG. 21, there is shown an example of such a generated Level Activation Table. This particular Level Activation Table represents a linearised form of the expression tree shown in FIG. 19. The Level Activation Table of FIG. 21 has a record for each node of the expression tree. These records each have the following fields "Index", "L Active", "R Active", "(L.andgate.{overscore (R)})", "({overscore (L)}.andgate.R)", "Leaf/Operator Entry", "Node Active", "Parent", "Node is L", "Generate L", "Generate R", "(L.andgate.R) op used", and "R Branch Index". The contents of the fields "Index", "(L.andgate.{overscore (R)})", "({overscore (L)}.andgate.R)", "Leaf/Operator Entry", "Parent", "Node is L", and "R Branch Index" are static in that they do not vary as a function of the current scanned pixel location. On the other hand, the contents of the fields "L Active", "R Active", "Node Active", "Generate L", "Generate R", and "(L.andgate.R) op used" may vary depending upon the currently scanned pixel location. The latter fields are updated by the Level Activation Table Update Module 504, for each group of pixel locations between the adjacent edges.

Detail Description Paragraph:
[0193] The "L Active" field is a logic field indicating whether the left branch of the binary node corresponding to the record is active or not depending upon the current group of scanned pixels. Similarly, the "R Active" field is a logic field indicating whether the right branch of the binary node corresponding to the record is active or not depending upon the current group of scanned pixels. During the initialisation phase there are no currently scanned pixels, thus all "L Active" and "R Active" fields of the binary nodes can be set to FALSE (0), with the possible exception of the "R Active" field of the root node. For example, the "R Active" field of the record corresponding to the root node 13 of FIG. 19 is set to TRUE (1), as the right branch of the node 13 is always active irrespective of the scanned pixel. It should be noted that as leaf nodes have no branches, there is no need to set the "L Active" and "R Active" fields for the corresponding records.

Detail Description Paragraph:
[0202] For example, the fields "Node Active" for binary nodes 8,11, and 12 shown in FIG. 19 are set to FALSE (0) during the initialisation phase as all the corresponding branches (L Active and R Active) are inactive (ie. FALSE). During the initialisation phase, the "Node Active" field for binary node 13 of FIG. 19 is set to TRUE (1) as its right branch is active (R Active=TRUE) and its left branch inactive (L Active=FALSE) and the region {overscore (L)}.andgate.R is required for the operation of the operator "over". It should be noted that as this field relates

to binary nodes only (ie compositing operations) there is no need to set these fields for records corresponding to leaf nodes. The "Node Active" field of a record indicates whether or not the corresponding binary operator is active.

Detail Description Paragraph:
[0208] For example, the fields "Generate L" for all the binary nodes 13, 12, 11, and 8 of FIG. 19 are set during the initialisation phase to FALSE (0) as this condition is not met by any of these nodes. Namely, the left branches of nodes 8, 11, 12, and 13 are all inactive ie L Active=FALSE (0). It should also be noted that as this field relates to binary nodes only (ie compositing operations) there is no need to set these fields for records corresponding to leaf nodes.

Detail Description Paragraph:
[0214] For example, the fields "Generate R" for the binary nodes 12, 11, and 8 of FIG. 19 are set during the initialisation phase to FALSE (0) as this condition is not met by any of these nodes. For instance, the right branches of nodes 8, 11, and 12 are all inactive ie R Active=FALSE (0). However, the field "Generate R" for the binary node 13 is set during the initialisation to TRUE (1) as the right branch is active, the left branch inactive and ({overscore (L)}.andgate.R)=TRUE. It should also be noted that as this field relates to binary nodes only (ie compositing operations) there is no need to set these fields for records corresponding to leaf nodes.

Detail Description Paragraph:
[0217] For example, as all the left branches of the binary nodes of FIG. 19 are inactive during the initialisation phase this field is set to FALSE (0) for all the binary nodes.

Detail Description Paragraph:
[0219] Returning now to FIG. 5, the Level Activation Table Generator Module 502 stores the Level Activation Table generated during the initialisation phase in the memory 34. The Update Module 504 then retrieves this initialised Level Activation Table for each group of scanned pixel locations between adjacent edges and updates the fields of the Table. The Update Module 504 changes the state of the fields "L Active", "R Active", "Node Active", "Generate L", "Generate R", and "(L.andgate.R) op used" depending upon the current group of scanned pixel locations. The Update Module 504 updates the records in a predetermined manner commencing at those records of the parent nodes of the leaf nodes corresponding to the active objects for that pixel location. Whenever the Update Module 504 changes the state of the "Node Active" field for a record, the Update Module 504 triggers a corresponding change in the state of the "L active" or "R active" field in the record of the parent node, depending on the state of the "Node is L" field. The Update Module 504 then updates the remaining fields of the record of the parent node in accordance with this newly changed "L Active" or "R Active field". This Updating process continues until a level is reached where the "Node Active" field remains unchanged. In the event there are more than one active objects, the Update Module updates the table one active object at a time.

Detail Description Paragraph:
[0220] The updating process of the Update Module 504 will now be explained with reference to FIG. 23, which shows an updated Level Activation Table for the expression tree of FIG. 20 where object C is active. The Update Module 504 first determines that the Object C is active for the current group of scanned pixel locations and then retrieves the initialised Level Activation Table from memory 34. The Update Module 504 determines the parent node of the active object C from the "Parent" field of the record of leaf node C. The update Module then proceeds to update the records of the table in the following manner commencing with the parent node of object C (record 8):

Detail Description Paragraph:

[0231] Returning now to FIG. 5, the Update Module 504 stores this updated Level Activation Table for the current group of scanned pixel locations in memory 34 which is then retrieved by the Traversal and Instruction Generator Module 506. The Traversal and Instruction generator Module 506 is a stack-machine based robot, which traverses the expression tree top-down, generating instructions in accordance with the updated Level Activation Table. The stack is used to store the "R branch Index" of a record. The stack-machine performs the following operations in accordance with the following pseudo-code:

Detail Description Paragraph:
[0253] The difference between a CLIP IN operation and a simple in operation is that the right branch operation is never performed. This can be seen from a comparison of the truth table of the IN operation as shown in FIG. 26 and from the truth table of the CLIP IN operation as shown in FIG. 27. In these truth tables, T stands for transparent (inactive), O stands for opaque, and L and R stand for L opacity unknown and R opacity unknown. The cases where a difference occurs are highlighted. In the CLIP IN case, activation of the right object enables the generation of instructions for the left object, such that the left object is edge clipped. Note that, the right object may be a composition of arbitrarily many objects, using whatever combinations and different fill rules as may be required.

Detail Description Paragraph:
[0255] The difference between a CLIP OUT operation and a simple out operation can be seen from a comparison of the truth table of the OUT operation as shown in FIG. 28 and from the truth table of the CLIP IN operation as shown in FIG. 29. Again, T stands for transparent (inactive), O stands for opaque, and L and R stand for L opacity unknown and R opacity unknown. The cases where a difference occurs are again highlighted. In the CLIP OUT case, activation of the right object prevents the generation of instructions for the left object, such that the left object is edge clipped. As for a CLIP IN, the right object may be an arbitrarily complex collection of objects, with different fill rules as required. It is only the activity state that is used for clipping.

Detail Description Paragraph:
[0283] The opacity tile module 612 interprets the supplied record as a fixed format record containing three 8-bit colour components, an 8-bit opacity value, an integer X phase, (px), a Y phase, (py), an X scale, (sx), a Y scale, (sy), and a 64 bit mask. These values originate in the display list generation and contained typically in the original page description. A bit address, a, in the bit mask, is determined by the formula:

CLAIMS:

1. A method of generating instructions for a directed adjacency graph, said directed adjacency graph comprising one or more parent nodes and one or more leaf nodes, each of which said parent node representing an operator and having branches to respective descendent nodes, and each of which said leaf node representing a graphic object, said method comprising the steps of: determining groups of one or more pixel locations; determining, for each said group, a portion of said directed adjacency graph in accordance with activities of the operators, wherein the said portion of the directed adjacency graph is that portion which passes data up the directed adjacency graph; and generating, for each said group, instructions for the determined portion of the directed adjacency graph, wherein operator instructions are generated for those operators of the determined portion of the directed adjacency graph having active branches and wherein leaf instructions are generated for those graphic objects which are active at said group of one or more pixel locations.

7. A method of generating instructions for an expression tree, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of

leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said method comprising the steps of: determining groups of one or more pixel locations; determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; traversing, for each said group, said expression tree, wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; and generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node.

9. A method of rendering an expression tree into a raster pixel image having a plurality of scanlines and a plurality of pixel locations on each said scanline, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said method comprising the steps of: generating a table representative of said expression tree, wherein said table comprises a plurality of records corresponding to associated said binary nodes and leaf nodes, and each said record of a said associated binary node comprises a first field indicating whether a said left region is required for operation of the operator of said associated binary node, a second field indicating whether a right region is required for operation of the operator of said associated binary node, a third field capable of indicating whether a said left branch of said associated binary node is active, and a fourth field capable of indicating whether a said right branch of said associated binary node is active; determining groups of one or more pixel locations; determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; updating, for each said group, said third and fourth fields of said table for said determined active and inactive branches; traversing, for each said group, said expression tree in accordance with said updated table wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node; and executing, for each said group, corresponding said generated instructions so as to render said image.

12. Apparatus for generating instructions for a directed adjacency graph, said directed adjacency graph comprising one or more parent nodes and one or more leaf nodes, each of which said parent node representing an operator and having branches to respective descendent nodes, and each of which said leaf node representing a graphic object, said apparatus comprising: means for determining groups of one or more pixel locations; means for determining, for each said group, a portion of said directed adjacency graph in accordance with activities of the operators, wherein the said portion of the directed adjacency graph is that portion which passes data up the directed adjacency graph; and means for generating, for each said group, instructions for the determined portion of the directed adjacency graph, wherein operator instructions are generated for those operators of the determined portion of the directed adjacency graph having active branches and wherein leaf instructions are generated for those graphic objects which are active at said group of one or more pixel locations.

18. Apparatus for generating instructions for an expression tree, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said apparatus comprising: means for determining groups of one or more pixel locations; means for determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; means for traversing, for each said group, said expression tree, wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; and means for generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node.

20. Apparatus for rendering an expression tree into a raster pixel image having a plurality of scanlines and a plurality of pixel locations on each said scanline, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said apparatus comprising: means for generating a table representative of said expression tree, wherein said table comprises a plurality of records corresponding to associated said binary nodes and leaf nodes, and each said record of a said associated binary node comprises a first field indicating whether a said left region is required for operation of the operator of said associated binary node, a second field indicating whether a right region is required for operation of the operator of said associated binary node, a third field capable of indicating whether a said left branch of said associated binary node is active, and a fourth field capable of indicating whether a said right branch of said associated binary node is active; means for determining groups of one or more pixel locations; means for determining, for each said group, whether the left and right branches of said one or more binary nodes are active or

inactive; means for updating, for each said group, said third and fourth fields of said table for said determined active and inactive branches; means for traversing, for each said group, said expression tree in accordance with said updated table wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; means for generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node; and means for executing, for each said group, corresponding said generated instructions so as to render said image.

23. A computer readable medium comprising a computer program for generating instructions for a directed adjacency graph, said directed adjacency graph comprising one or more parent nodes and one or more leaf nodes, each of which said parent node representing an operator and having branches to respective descendent nodes, and each of which said leaf node representing a graphic object, said computer program comprising: code for determining groups of one or more pixel locations; code for determining, for each said group, a portion of said directed adjacency graph in accordance with activities of the operators, wherein the said portion of the directed adjacency graph is that portion which passes data up the directed adjacency graph; and code for generating, for each said group, instructions for the determined portion of the directed adjacency graph, wherein operator instructions are generated for those operators of the determined portion of the directed adjacency graph having active branches and wherein leaf instructions are generated for those graphic objects which are active at said group of one or more pixel locations.

29. A computer readable medium comprising a computer program for generating instructions for an expression tree, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said computer program comprising: code for determining groups of one or more pixel locations; code for determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; code for traversing, for each said group, said expression tree, wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; and code for generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node.

31. A computer readable medium comprising a computer program for rendering an expression tree into a raster pixel image having a plurality of scanlines and a plurality of pixel locations on each said scanline, said expression tree having a plurality of nodes comprising one or more binary nodes and a plurality of leaf nodes, wherein each said binary node having a left branch to a descendent said node and a right branch to a descendent said node and representing a binary operation on said two descendant nodes, and wherein each said node represents a graphic object, with one or more said graphic objects overlapping, each said overlapping graphics objects comprising a left node region, a common region, and a right node region, said computer program comprising: code for generating a table representative of said expression tree, wherein said table comprises a plurality of records corresponding to associated said binary nodes and leaf nodes, and each said record of a said associated binary node comprises a first field indicating whether a said left region is required for operation of the operator of said associated binary node, a second field indicating whether a right region is required for operation of the operator of said associated binary node, a third field capable of indicating whether a said left branch of said associated binary node is active, and a fourth field capable of indicating whether a said right branch of said associated binary node is active; code for determining groups of one or more pixel locations; code for determining, for each said group, whether the left and right branches of said one or more binary nodes are active or inactive; code for updating, for each said group, said third and fourth fields of said table for said determined active and inactive branches; code for traversing, for each said group, said expression tree in accordance with said updated table wherein the left branch of any previously traversed said binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said left node region is required for the binary operation of said previously traversed binary node and the left branch is active and the right branch is inactive of said previously traversed binary node, and wherein a right branch of any previously traversed binary node is traversed to its said descendent node if the right and left branches of said previously traversed binary node are active or if a said right node region is required for the binary operation of said previously traversed binary node and the right branch is active and the left branch is inactive of said previously traversed binary node; code for generating, for each said group, operator instructions for any said traversed binary node having active said right and left branches, and leaf value instructions for any traversed leaf node; and code for executing, for each said group, corresponding said generated instructions so as to render said image.